



Fast recognition of a Digital Straight Line subsegment: Two algorithms of logarithmic time complexity

Isabelle Sivignon

► To cite this version:

Isabelle Sivignon. Fast recognition of a Digital Straight Line subsegment: Two algorithms of logarithmic time complexity. Discrete Applied Mathematics, 2015, pp.130-146. 10.1016/j.dam.2014.04.017 . hal-01120578

HAL Id: hal-01120578

<https://hal.science/hal-01120578>

Submitted on 26 Feb 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Fast recognition of a Digital Straight Line Subsegment: two algorithms of logarithmic time complexity

Isabelle Sivignon¹

GIPSA-lab, CNRS, UMR 5216, F-38420, France

Abstract

Given a Digital Straight Line (DSL) of known characteristics (a, b, μ) , we address the problem of computing the characteristics of any of its subsegments. We propose two new algorithms that use the fact that a digital straight segment (DSS) can be defined by its set of separating lines. The representation of this set in the \mathbb{Z}^2 space leads to a first algorithm of logarithmic time complexity. This algorithm precises and extends existing results for DSS recognition algorithms. The other algorithm uses the dual representation of the set of separating lines. It consists of a smart walk in the so called Farey Fan, which can be seen as the representation of all the possible sets of separating lines for DSSs. Indeed, we take profit of the fact that the Farey Fan of order n represents in a certain way all the digital segments of length n . The computation of the characteristics of a DSL subsegment is then equivalent to the localization of a point in the Farey Fan. Using fine arithmetical properties of the fan, we design a fast algorithm of theoretical complexity $\mathcal{O}(\log(n))$ where n is the length of the subsegment. Experiments show that our algorithms are also efficient in practice, with a comparison to the ones previously proposed by Lachaud and Said [1]: in particular, the second one is faster in the case of “small” segments.

Keywords: Digital geometry, Digital straight segment recognition, subsegment, local convex hull, Farey fan

1. Introduction

Digital Straight Lines (DSL) and Digital Straight Segments (DSS) have been known for many years to be interesting tools for digital curve and shape analysis. The applications range from simple coding to complex multiresolution analysis and geometric estimators (see for instance [2] for a recent example). All these applications require to solve the so called DSS recognition problem. Many algorithms, using arithmetics [3], combinatorics [4] or dual-space [5] have been

Email address: `isabelle.sivignon@gipsa-lab.grenoble-inp.fr` (Isabelle Sivignon)

¹This work has been partially supported by the French *Agence Nationale de la Recherche* (Grant agreement ANR-11-BS02-009)

8 proposed to solve this problem, reaching a computational complexity of $\mathcal{O}(n)$
9 for a DSS of length n (see also [6] for an overview).

10 When no further information is known, all these algorithms are actually
11 optimal. They at the same time decide if the set of grid points is a DSS and
12 compute its characteristics (minimal in some sense). However, we sometimes
13 know beforehand that the set of grid points is a DSS: the algorithm does not
14 need to decide anymore and we can hope for a sublinear-in-time recognition
15 algorithm. For instance, this extra information may come from the knowledge
16 of the characteristics of a DSL containing the set of grid points. This occurs
17 for example in [7] where the multiresolution geometry of a digital object is
18 considered. Another example concerns the digitization of a straight segment
19 on a grid of given size: we know that the set of grid points is a DSS, but its
20 characteristics may be much smaller than the ones of the input straight segment
21 (and not greater than the grid size).

22 In [7], the authors introduce the following problem: given a DSL of known
23 characteristics and a subsegment of this DSL, compute the minimal characteris-
24 tics of the DSS. The authors present two algorithms (SmartDSS and ReversedS-
25 martDSS) to solve this problem in [7, 8, 1]: both use the decomposition into
26 continuous fractions of the DSL slope and reach a logarithmic complexity.

27 However, a deeper search in the state-of-the-art shows that this problem
28 is not so new. Indeed, in [9], the author presents a quick sketch of a method
29 that solves it using the Farey Fan. The announced complexity of the method
30 is $\mathcal{O}(\log^2 n)$ for a segment of length n . Much later, the authors of [10], try
31 to compute the “reduction” of a straight line, which is a simplification of DSL
32 characteristics over a bounded domain. As we will see, this reduction does not
33 compute the minimal characteristics, but the idea is similar.

34 Our contribution in this paper is to demonstrate that it is possible to solve
35 the DSL subsegment problem in logarithmic time complexity revisiting and
36 deepening the study of the two state-of-the-art algorithms [9] and [10].

37 The first algorithm is based on the local convex hull algorithm developed in
38 [10] together with the framework for DSS recognition described in [11], but we
39 provide the theoretical results which enable to efficiently compute the minimal
40 characteristics of a subsegment from this hull. The second algorithm, detailed
41 in Section brings the method introduced in [9] up to date: we investigate it
42 further to provide a thoroughly defined algorithm. Moreover, we show how
43 its complexity can be lowered to $\mathcal{O}(\log(n))$ with an astute use of arithmetical
44 properties of the Farey Fan. The latter algorithm was first presented in [12] and
45 a more detailed description is provided here.

46 Section 2 is dedicated to the presentation of the notions used in this work.
47 Section 3 describes the algorithm based on the local convex hull computation.
48 The algorithm using the dual representation, and more specifically the Farey
49 Fan is detailed in Section 4. At the end of this section, two extensions for
50 slightly different frameworks are presented : in particular, we show that the
51 second algorithm can be directly and reliably applied when the input data is
52 not a DSL but a straight line with non integer parameters. The last section is
53 classically devoted to experimental validations.

54 2. Preliminary definitions

55 2.1. Digital line, segment and minimal characteristics

56 A *Digital Straight Line* (DSL for short) of integer characteristics $(a, b, \mu) \in$
57 \mathbb{Z}^3 is the infinite set of digital points $(x, y) \in \mathbb{Z}^2$ such that $0 \leq ax - by + \mu <$
58 $\max(|a|, |b|)$, with a and b relatively prime [3]. These DSL are 8-connected
59 and often called *naive*. The slope of the DSL is the fraction $\frac{a}{b}$ and $\frac{\mu}{b}$ is the
60 shift at the origin. In the following, without loss of generality, we assume that
61 $0 \leq a \leq b$, such that, on a DSL, there is exactly one pixel for each value of x . In
62 this context, it is easy to see that the set of pixels of a given DSL is defined by a
63 unique triplet (a, b, μ) . The remainder of a DSL of characteristics (a, b, μ) for a
64 given digital point (x, y) is the value $ax - by + \mu$. The *upper (resp. lower) leaning*
65 *line* of a DSL is the straight line $ax - by + \mu = 0$ (resp. $ax - by + \mu = b - 1$).
66 Upper (resp. lower) leaning points are the digital points of the DSL lying on
67 the upper (resp. lower) leaning lines.

68 A *Digital Straight Segment* (DSS) is a finite 8-connected part of a DSL. It
69 can be defined by the characteristics of a DSL containing it and two endpoints
70 P and Q . However, a DSS belongs to an infinite number of DSLs. In this
71 context, the *minimal characteristics* of a DSS are the characteristics of the DSL
72 containing it with minimal b [13, 1]. Since a DSL is defined by a unique triplet
73 (a, b, μ) , the values of a and μ are uniquely defined for a given b , and the minimal
74 characteristics of a DSS are also uniquely defined.

75 **Definition 1 (minimal characteristics).** Let $\mathcal{L} = \{(a, b, \mu) \in \mathbb{Z}^3, 0 \leq a \leq b,$
76 $\gcd(a, b) = 1\}$. For a given DSS S , we can define $\mathcal{L}_S = \{L \in \mathcal{L}, \text{the pixels of } S \text{ all}$
77 $\text{belong to the DSL of characteristics } L\}$. Let $f : \mathcal{L} \rightarrow \mathbb{Z}, f(a, b, \mu) = b$. Then
78 the *minimal characteristics* of S is the triplet $(a_S, b_S, \mu_S) = \arg \min_{L \in \mathcal{L}_S} f(L)$.

79 Note that the notions of leaning points and lines are similarly defined for
80 DSSs. DSS recognition algorithms aim at computing the minimal character-
81 istics of a DSS, taking profit of the following fact: (a, b, μ) are the minimal
82 characteristics of a DSS if and only if the DSS contains at least three leaning
83 points [3]. In this case, the minimal characteristics are the characteristics of the
84 DSS upper leaning line.

85 2.2. Minimal characteristics, separating lines and dual space

86 If we consider the digitization process related to this DSL definition, the
87 points of the DSL \mathbf{L} of parameters (a, b, μ) are simply the grid points (x, y)
88 lying below or on the straight line $l : ax - by + \mu = 0$ (Object Boundary
89 Quantization), and such that the points $(x, y + 1)$ lie above l . We say that \mathbf{L} is
90 the digitization of the straight line l . Note that \mathbf{L} is also the digitization of all
91 the straight lines of equation $ax - by + \rho = 0$ with $\mu \leq \rho < \mu + 1$, where $\rho \in \mathbb{R}$.
92 These lines separate the points X of the DSL from the points $X + (0, 1)$, denoted
93 by \overline{X} (as in [11]), and they are called *separating lines*. Figure 1(a) illustrates
94 the separating lines of a DSL.

95 A similar set of lines can be defined if a DSS is considered. Let us denote
 96 by X the points of the DSS and by \bar{X} the points of the DSS translated by the
 97 vector $(0, 1)$. The separating lines are the lines which are above the upper part
 98 of the convex hull (upper convex hull for short) of the points X and strictly
 99 below the lower part of the convex hull (lower convex hull for short) of the
 100 points \bar{X} (see Figure 1 for an illustration). Note that the strict constraint on
 101 the lower convex hull makes this definition slightly different from the classical
 102 definition in computational geometry. However, geometrically speaking, the set
 103 of separating lines is also bounded by the critical support lines of the two convex
 104 hulls. We actually have the property that an 8-connected set of grid points is a
 105 DSS if and only if its set of separating lines is not empty. This property is used
 106 in [11, 14] to design a fast linear in time DSS recognition algorithm. Among
 107 the separating lines, the line with integer characteristics (a, b, μ) with minimal b
 108 and minimal μ defines the minimal characteristics of the DSS. In the algorithm
 109 of [11], the points of the DSS are added one by one and the set of separating
 110 lines is updated accordingly, but the minimal characteristics are not extracted.

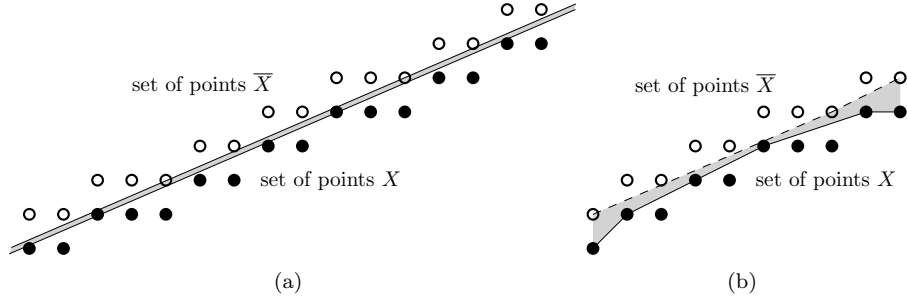


Figure 1: The separating lines of a DSL(a) and of a DSS (b) are the straight lines lying in the gray area.

111 For a DSS of minimal characteristics (a, b, μ) , the structure of the set of
 112 separating lines is perfectly known. Indeed, it is defined by the leaning points of
 113 the DSS. If U_f (resp. U_l) is the leftmost (resp. rightmost) upper leaning point,
 114 and L_f (resp. L_l) the leftmost (resp. rightmost) leaning point, then the set of
 115 separating lines is bounded by the lines $(U_f, L_l + (0, 1))$ and $(L_f + (0, 1), U_l)$ which
 116 are the critical support lines, and the lines (U_f, U_l) and $(L_f + (0, 1), L_l + (0, 1))$
 117 which are edges of the lower and upper convex hulls respectively. Figure 2(a)
 118 illustrates this structure.

119 The set of separating lines can also be defined in a dual space, also called
 120 parameter space. In this space a straight line $l : \alpha x - y + \beta = 0$ is represented
 121 by the 2D point (α, β) .

122 Given a DSS S , a line $l : \alpha x - y + \beta = 0$ is a separating line if and only if for
 123 all $(x, y) \in S$, $0 \leq \alpha x - y + \beta < 1$. This definition is strictly equivalent to the
 124 one given previously. The preimage of S is the representation of the separating
 125 lines in the dual space and is defined as $\mathcal{P}(S) = \{(\alpha, \beta), 0 \leq \alpha \leq 1, 0 \leq \beta \leq$
 126 $1; \mid \forall (x, y) \in S, 0 \leq \alpha x - y + \beta < 1\}$. As in [9], let us define a ray in this space.

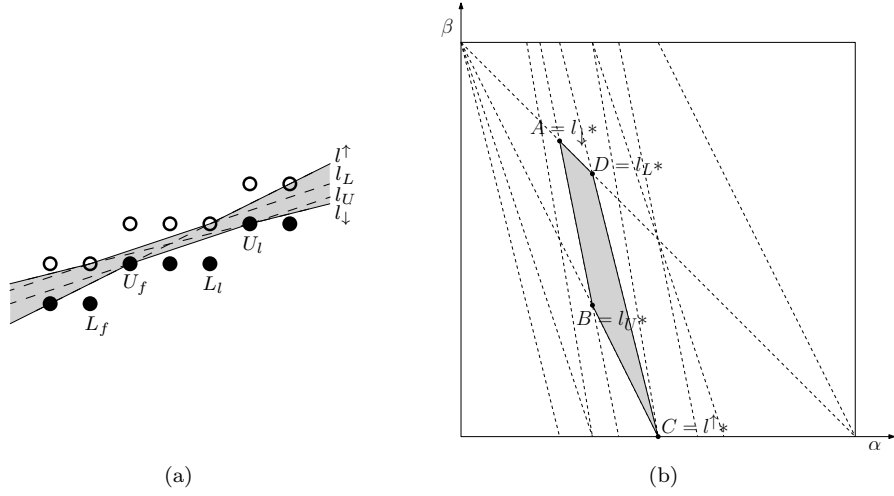


Figure 2: (a) DSS of minimal characteristics $(1, 3, 1)$ with its leaning points U_f, U_l, L_f, L_l . (b) Each vertex of the preimage $\mathcal{P}(S)$ maps to a straight line in the digital space. The vertex $B(\frac{1}{3}, \frac{1}{3})$ maps to the upper leaning line, the characteristics of which are the minimal characteristics of the DSS.

Definition 2 (Ray). Let x and y be two integers. The *ray* defined by x and y is defined and denoted as follows:

$$R(x, y) = \{(\alpha, \beta) | \beta = -x\alpha + y\}$$

127 x is called the *slope* of the ray.

128 Note that x is not the geometrical slope of the ray but its absolute value. In
129 the following, the order on the slopes is to be understood as the order on the
130 absolute values of the geometrical slopes.

131 Given a DSS, any point (x, y) of the DSS induces two linear constraints on
132 the preimage: $\alpha x - y + \beta - 1 < 0$ and $\alpha x + \beta - y \geq 0$. In other words, the
133 preimage is bounded by two parallel rays: it is below the ray $R(x, y + 1)$ and
134 above the ray $R(x, y)$.

135 This definition enables to prove that the preimage of a DSS is a convex
136 polygon with a well-defined structure that is directly related to the leaning
137 points and lines defined by its minimal characteristics [9, 15]. Figure 2 from
138 [16] illustrates this point. In Figure 2(b), all the rays induced by the DSS pixels
139 are depicted as dotted lines. The preimage (in gray) is the intersection of all the
140 constraints bounded by the rays. It is for instance well known that the edges
141 of a DSS preimage are segments of rays induced by the first and last lower and
142 upper DSS leaning points. As a consequence, the preimage of a DSS has either
143 three or four edges. Proposition 1 recalls in detail this specific structure.

144 **Proposition 1 ([16]).** Let $\mathcal{P}(S)$ be the preimage of S . Let $ABCD$ be the poly-
145 gon defined by this preimage, where A is the upper left most vertex, and the

vertices are named counterclockwise. Following the notations of Figure 2 we have :

- The vertex $B = l_U^*$ maps to the upper leaning line (U_f, U_l) ;
- The vertex $D = l_L^*$ maps to the lower leaning line (L_f, L_l) translated by the vector $(0, 1)$ in the digital space ;
- The vertex $A = l_\downarrow^*$ maps to the straight line $(U_l, L_f + (0, 1))$;
- The vertex $C = l_\uparrow^*$ maps to the straight line $(U_f, L_l + (0, 1))$.

Note that point B or D may be on the line (AC) . (a, b, μ) are the minimal characteristics of S if and only if $B = l_U^* = (\frac{a}{b}, \frac{\mu}{b})$ (a and b relatively prime). B is called the **characteristic point** of $\mathcal{P}(S)$.

In the rest of the paper, and especially in Section 4, the edges $[AB]$ and $[BC]$ are called **lower edges**. This dual representation of the set of separating lines has also been used to design DSS recognition algorithms with a linear time complexity [5, 17].

2.3. Statement of the problem

Consider now the following problem:

Problem 1. Given a DSL \mathbf{L} of characteristics $(a_{\mathbf{L}}, b_{\mathbf{L}}, \mu_{\mathbf{L}})$ and two points $P(x_P, y_P)$ and $Q(x_Q, y_Q)$ of this DSL (with $x_P < x_Q$), compute the minimal characteristics (a, b, μ) of the DSS $S = \{(x, y) \in \mathbf{L} \mid x_P \leq x \leq x_Q\}$.

Some easy cases can be withdrawn rapidly (see [8, 1]): if $x_Q - x_P \geq 2b_{\mathbf{L}}$, then the DSS contains at least three leaning points of the DSL and the minimal characteristics of the DSS are simply equal to $(a_{\mathbf{L}}, b_{\mathbf{L}}, \mu_{\mathbf{L}})$.

For the general case, classical DSS recognition algorithm can obviously be used. But the best complexity of such algorithms is linear in the number of pixels of the DSS. The aim here is to take profit of the extra information given by the DSL that contains the DSS to design a sublinear algorithm. In [8, 1] the authors describe two algorithms of logarithmic time complexity to solve this problem using continued fractions and a top-down or bottom-up path in the Stern-Brocot tree (see for instance [18]). In the following sections, we present two new algorithms of logarithmic time complexity to solve this problem. They use the two representations of the set of separating lines presented in Section 2.2 and illustrated in Figure 2. The algorithm of Section 3 uses properties of the upper and lower local convex hulls (Figure 2(a)) to compute the minimal characteristics, while the algorithm presented in Section 4 takes profit of a strong arithmetical structure (in the dual space, see Figure 2(b)) called Farey fan to solve the problem. The experimental section will show that our algorithms, especially the second one, have a very nice behaviour.

3. Fast computation of local convex hulls

3.1. Rewriting of the problem

Problem 1 can be rewritten as follows:

Problem 2. Given a separating line l of characteristics (a_l, b_l, μ_l) and two bounding abscissas x_P and x_Q such that $x_P < x_Q$, find the line of minimal characteristics that is separating for the same set of points as l for the grid points between x_P and x_Q .

In the following, we denote by X the grid points below l on the interval $[x_P, x_Q]$ such that the points $X + (0, 1)$, denoted by \bar{X} are above l . The points X form a DSS by definition.

We present here two properties that show that the minimal characteristics of the DSS can be retrieved from some particular edges of the lower convex hull of \bar{X} and the upper convex hull of X .

Property 1. Consider the remainder function $r(\alpha, \beta)(x, y) = \alpha x - y + \beta$. Let l be a line of slope α and intercept β that is separating for the DSS S of minimal characteristics (a, b, μ) . If the points of S are denoted by X , then the function $r(\alpha, \beta)$ restricted over the points X reaches its smallest positive value for an upper leaning point of S . Similarly, the function $r(\alpha, \beta)$ reaches its greatest value for a lower leaning point of S . Equivalently, if $r(\alpha, \beta)$ is restricted to the points \bar{X} , it reaches its greatest negative value for a point $L + (0, 1)$ where L is a lower leaning point of S , and smallest value for a point $U + (0, 1)$ where U is an upper leaning point of S .

PROOF. If $\alpha = \frac{a}{b}$, the result is straightforward. Otherwise, if $\alpha > \frac{a}{b}$, then l can be written as a linear combination of l^\uparrow and a line l_0 of slope $\alpha_0 = \frac{a}{b}$ and intercept β_0 , lying in between l_U and l_L . This is easy to see in the dual space representation and illustrated in Figure 3. If $l = (1 - t)l_0 + tl^\uparrow$ where $t \in [0, 1]$, the remainder function $r(\alpha, \beta)$ is equal to $(1 - t)r(\alpha_0, \beta_0) + tr(\alpha^\uparrow, \beta^\uparrow)$. The smallest positive value of $r(\alpha^\uparrow, \beta^\uparrow)$ is equal to 0 and reached for the point U_f . Similarly, the smallest positive value of $r(\alpha_0 = \frac{a}{b}, \beta_0)$ is reached for all the upper leaning points of S , thus for U_f . All in all, the smallest positive value of $r(\alpha, \beta)$ is reached for the point U_f . In the same way, the greatest value of $r(\alpha_0 = \frac{a}{b}, \beta_0)$ is reached for all the lower leaning points of S , and the greatest value of $r(\alpha^\uparrow, \beta^\uparrow)$ is reached for the point L' , which concludes the proof when α is greater than $\frac{a}{b}$. The case $\alpha < \frac{a}{b}$ is similar, replacing the line l^\uparrow by the line l_\downarrow .

From this property, we deduce that the grid point closest to l and below or on l is an upper leaning point for the DSS minimal characteristics. Similarly, the grid point closest to l and above l is the translation by $(0, 1)$ of a lower leaning point for the DSS minimal characteristics. These two points are denoted by U and L respectively.

Consider now the upper convex hull of X and the lower convex hull of \bar{X} . Then from 2.2, Proposition 1 and Property 1, the DSS minimal characteristics

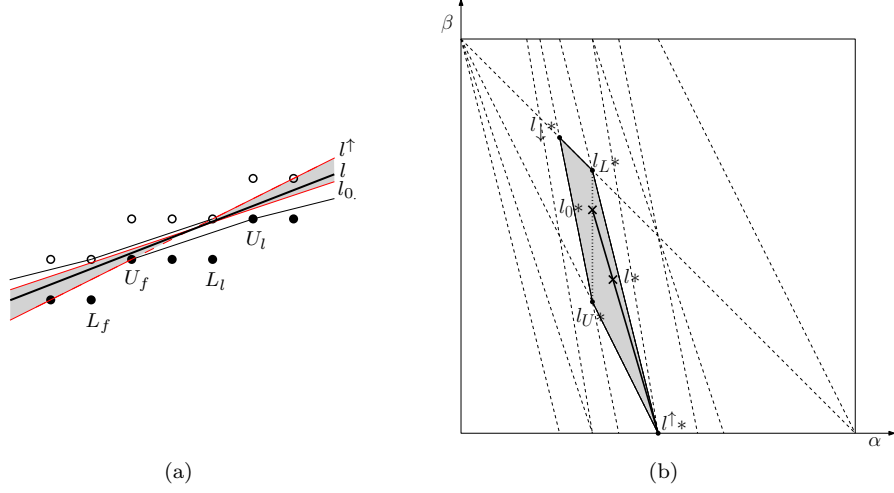


Figure 3: A separating line l of slope greater than $\frac{a}{b}$ can be written as a linear combination of l^\uparrow and a line l_0 .

are given by an edge passing through U or L . This leaves us with four edges to check, and the following property is used to conclude.

Property 2. Consider the two edges e_0 and e_1 of the upper convex hull of X passing through U , and the two edges e_2 and e_3 of the lower convex hull of \bar{X} passing through $L + (0, 1)$. We denote by $\frac{a_i}{b_i}$ (with $\gcd(a_i, b_i) = 1$) the slopes of these edges. (a, b, μ) are the minimal characteristics of the DSS S if and only if $(a, b) = (a_k, b_k)$ where $b_k = \max(b_i)$, and μ is such that the remainder of the DSS (a, b, μ) is equal to 0 on U .

PROOF. From Proposition 1, each edge e_i lies on a line that links either (i) two leaning points, both upper or both lower translated by $(0, 1)$, of the DSS we are looking for, or (ii) one upper leaning point to the translation by $(0, 1)$ of a lower leaning point. The minimal characteristics (a, b) are given by an edge of type (i). Moreover, at most two out of the four edges may lie on the same line, and since there is at most two edges of type (ii), at least one edge is of type (i). If two edges are of type (i), they have the same slope. Since we have no information on the points of the edges apart from U and L , the slopes are used to discriminate between edges of type (i) and edges of type (ii). Indeed, by definition, an edge of type (ii) has a shorter period, i.e. a smallest b_i than an edge of type (i), which concludes the first part of the proof. Once the characteristics (a, b) are known, we just use the fact that U is an upper leaning point of the DSS we are looking for to compute μ and thus end the proof.

3.2. Algorithm

Given a line l and two bounds x_P and x_Q , Properties 1 and 2 directly lead to an algorithm to solve Problem 2 in three steps:

- 248 1. compute the upper hull of the grid points X - keep the point closest to l
249 and its two neighbour edges;
- 250 2. compute the lower hull of the grid points \bar{X} - keep the point closest to l
251 and its two neighbour edges;
- 252 3. among the four edges of slope $\frac{a_i}{b_i}$, the one with maximal b_i gives the
253 solution.

254 The algorithm of [10] offers a fast solution to solve the first two steps. Indeed,
255 the authors propose an algorithm of complexity $\mathcal{O}(\log(x_Q - x_P))$ to compute the
256 upper and the lower convex hull of the grid points below and above a straight
257 line and between a minimal and maximal abscissa.

258 In this article, the authors actually mention an algorithm to compute the
259 “reduction” of a straight line over a domain defined by a minimal and a maximal
260 abscissa. The aim is to reduce the coefficients of a straight line digitized over
261 a finite domain. This algorithm is based on the computation of the critical
262 support lines of the convex hulls computed by the algorithm. However, the
263 “reduction” they compute is not equal to the line of minimal characteristics.
264 The first reason is because, as we saw, the critical support lines contain a point
265 of \bar{X} . The second reason is given hereafter and illustrated in Figure 4.

266 If there is no grid point lying exactly on l in the interval $[x_P, x_Q]$ this algo-
267 rithm computes exactly the hulls we look for (Figure 4 (a)). However, if there
268 exists a point R on l of abscissa $x_P \leq x_R \leq x_Q$, then the lower convex hull
269 computed by this algorithm is erroneous for our purpose: indeed, the computed
270 lower convex hull of \bar{X} contains the point R which is a point of X and not a
271 point of \bar{X} (Figure 4 (b)). To solve this problem, we use the fact that, for a
272 line $l : a_{\mathbf{L}}x - b_{\mathbf{L}}y + \mu_{\mathbf{L}} = 0$ with integer characteristics, there is no grid point
273 lying strictly between l and the line $l' : a_{\mathbf{L}}x - b_{\mathbf{L}}y + \mu_{\mathbf{L}} = 1$. The trick is thus
274 to compute the lower convex hull of the points \bar{X} defined by the line l' .

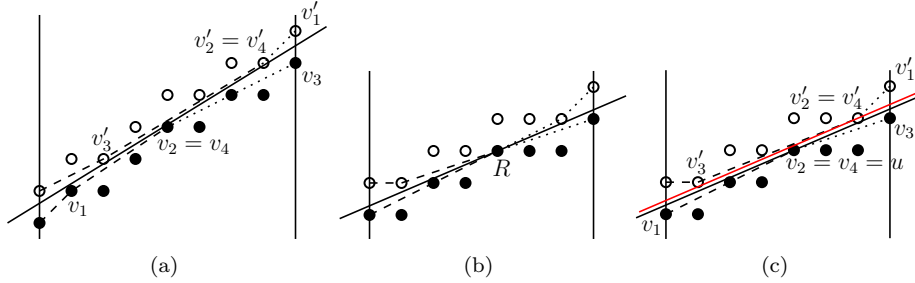


Figure 4: (a) and (b) Illustration of the algorithm of [10]. In (b), the lower convex hull of \bar{X} contains a point of X . (c) Modification so that the lower convex hull of \bar{X} does not contain a point of X .

275 As in [10], the upper convex hull of the points X defined by a line l is
276 split in two parts: the left part is composed of the edges of slope greater than
277 the slope of l , while the right part is composed of the edges of slope lower
278 than the slope of l . Then, the computation of the upper convex hull of the

279 points X is completed in two steps, one for each part of the convex hull. Let
 280 $\text{localCHLeftRight}(a_L, b_L, \mu_L, n)$ be Algorithm 1 of [10] for a line $l : a_L x - b_L y +$
 281 $\mu_L = 0$ and abscissas in between 0 and n . This algorithm correctly returns the
 282 left part of the upper convex hull of the points X . From this convex hull, we
 283 only keep the last two points, the last one being the closest to l .

Algorithm 1: $\text{upperConvexHullOfX}(a_L, b_L, \mu_L, P, Q)$

```

1  $v_1, v_2 = \text{localCHLeftRight}(a_L, b_L, r_P, x_Q - x_P)$ 
   $v_1 = v_1 + P, v_2 = v_2 + P$ 
2  $v_3, v_4 = \text{localCHLeftRight}(a_L, b_L, b_L - r_Q, x_Q - x_P)$ 
   $v_3 = Q + (0, 1) - v_3, v_4 = Q + (0, 1) - v_4$ 
3 The closest point  $u$  is equal to  $v_2$  or  $v_4$ 
  Return  $V = (v_1, v_2, v_3, v_4)$  removing multiple copies, and closest point  $u$ 

```

Algorithm 2: $\text{localCHDSLSubsegment}(a_L, b_L, \mu_L, P, Q)$

```

1  $V_{inf}, u = \text{upperConvexHullOfX}(a_L, b_L, \mu_L, P, Q)$ 
2  $V_{sup} = \text{upperConvexHullOfX}(a_L, b_L, b_L - r_Q - 1, (0, 0), Q - P)$ 
  forall  $v$  in  $V_{sup}$  do  $v = Q + (0, 1) - v$ 
  Let  $E$  be the set of edges  $e$ , where  $e = (v_i, v_{i+1})$  with  $v_i \in V_{inf}$  or
   $v_i \in V_{sup}$ 
   $(b, a) = (0, 0)$ 
3 forall  $e(e_x, e_y) \in E$  do
  if  $e_x > b$  then  $(b, a) = e$ 
  end
4  $\mu = -au_x + bu_y$ 
  Return  $(a, b, \mu)$ 

```

284 Algorithm 1 computes the upper convex hull of the points X lying right below
 285 a line l between two points P and Q . r_P (resp. r_Q) stands for the remainder
 286 of point P (resp. Q) for the characteristics (a_L, b_L, μ_L) . It consists of two calls
 287 to localCHLeftRight on lines 1 and 2. The call on line 2 corresponds to the
 288 computation of the hull from right to left. It returns the ordered list of vertices
 289 adjacent to the two edges defined in Property 2 and also remembers the closest
 290 point denoted by u on line 3.

291 Algorithm 2 is the general algorithm to solve Problem 1, rewritten as Prob-
 292 lem 2. Lines 1 and 2 compute the upper hull of the points of X and the
 293 lower hull of the points of \bar{X} respectively. The result of these calls is illus-
 294 trated in Figure 4(a) and (c): points v_i are the one returned by the first call to
 295 $\text{upperConvexHullOfX}$ on line 1, whereas the points v'_i are the one returned by
 296 the second call on line 2, using the trick presented in the previous paragraph.
 297 The loop on line 3 corresponds to the application of Property 2: the edge with
 298 maximal x -coordinate defines the sought characteristics. The point u is the point
 299 closest to l below l : from Property 1 u is an upper leaning point of the DSS we

300 are looking for and μ is computed from this point on line 4

301 Concerning the time complexity, algorithm `localCHLeftRight`(a_L, b_L, μ_L, n)
 302 works in $\mathcal{O}(\log(n))$ from [10], and is called four times. All other operations are
 303 done in constant time, so that the complexity of Algorithm `localCHDSLSubsegment`
 304 is $\mathcal{O}(\log(n))$ for a subsegment of length n .

305 4. Fast walk in the Farey fan

306 This section is dedicated to the presentation of another algorithm to solve
 307 Problem 1 using the dual space representation of the set of separating lines.

308 4.1. Rewriting of the problem

309 Let us consider all the possible rays $R(x, y)$ as defined in Section 2 with
 310 $0 < y \leq x \leq n$. This is equivalent to considering all the linear constraints
 311 induced by all the pixels (x, y) such that $0 < y \leq x \leq n$.

312 **Definition 3 (Farey Fan).** The Farey Fan of order n , denoted by \mathcal{F}_n is defined
 313 in the (α, β) space as the arrangement of all the rays $R(x, y)$ such that
 314 $0 \leq y \leq x \leq n$, and such that $0 \leq \alpha \leq 1$ and $0 \leq \beta \leq 1$.

315 A *facet* of \mathcal{F}_n is a cell of dimension 2 of this arrangement. In the following, a
 316 *point* of \mathcal{F}_n stands for any point v of the (α, β) space ($0 \leq \alpha \leq 1$ and $0 \leq \beta \leq 1$)
 317 belonging to a ray, and such that the abscissa of v is a fraction of denominator
 318 smaller than or equal to n .

319 If P and Q are respectively the first and last point of the DSS, from the
 320 definition and the previous remarks, key Property 3 follows.

321 **Property 3 ([9]).** For any n , there is a bijection between the facets of \mathcal{F}_n
 322 and the DSSs of length n (composed of $n + 1$ pixels) such that $P = (0, 0)$ and
 323 $Q = (n, y_q)$ with $0 < y_q \leq n$.

324 **Definition 4.** Let S be a DSS of length n . $Facet(S)$ is the facet equal to $\mathcal{P}(S)$
 325 in the Farey fan of order n .

326 Moreover, from Proposition 1 a one-to-one correspondence can be defined
 327 between a facet and the characteristic point of the facet.

328 **Definition 5.** Let f be a facet of the Farey fan of order n . $CPoint(f)$ is the
 329 point v of f such that if $v = (\frac{p}{q}, \frac{r}{q})$, then (p, q, r) are the minimal characteristics
 330 of the DSS $Facet^{-1}(CPoint^{-1}(v))$.

331 The Farey Fan of order 6 is depicted in Figure 5(a). The characteristic point
 332 of a few facets is depicted. Note that three types of facets can be identified :

- 333 • quadrilateral facets (denoted by Q , in orange in Figure 5(a)) ;
- 334 • upper triangular facets (denoted by T_{\uparrow} , in green in Figure 5(a)) ;

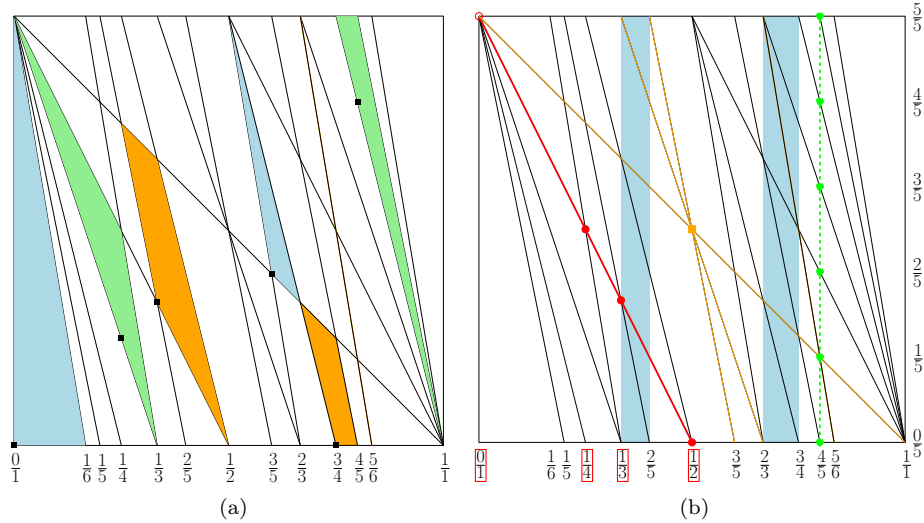


Figure 5: (a) The Farey Fan of order 6. (b) Illustration of properties 4 to 6.

• lower triangular facets (denoted by T_{\downarrow} , in blue in Figure 5(a)).

Let us now go back to Problem 1. After a translation of the characteristics of \mathbf{L} such that P is set to the origin ($\mu_{\mathbf{L}} \leftarrow \mu_{\mathbf{L}} + a_{\mathbf{L}}x_P - b_{\mathbf{L}}y_P$), this problem is equivalent to the following one :

Problem 3. Given a point $\mathbf{\Lambda}(\frac{a}{b}, \frac{\mu}{b})$, find the point v of the Farey fan of order $n = x_Q - x_P$ such that $\mathbf{\Lambda} \in CPoint^{-1}(v)$.

In other words, the problem is to find the characteristic point of the facet of \mathcal{F}_n containing $\mathbf{\Lambda}$.

All in all solving Problem 3 is equivalent to performing a point location in an arrangement of lines. However, the number of facets in the Farey fan of order n (which is equal to the number of DSS of length n) is in $\mathcal{O}(n^3)$ [19, 20, 21], and point location algorithms in such a structure are expensive in term of both time and space complexity [22]. This brute force approach is then less efficient than classical DSS recognition algorithms [3, 4, 23, 5].

In the following sections, we revisit the approach proposed by [9] and present an algorithm to solve Problem 3 in time complexity $\mathcal{O}(\log n)$, without explicitly computing the Farey fan. In the next section, we recall several structural and arithmetical properties of the Farey fan, and derive some very useful corollaries. These properties are the core of the algorithm detailed in section 4.3.

4.2. Properties of the Farey Fan

The Farey series of order n is the set of irreducible fractions in $[0, 1]$ of denominator lower than or equal to n [24]. The construction of the Farey

series of order n from the Farey series of order $n - 1$ is simply done as follows.
 Consider two consecutive fractions $\frac{p}{q}$ and $\frac{p'}{q'}$ of the Farey series of order n . All
 the properties below are illustrated in Figure 5(b) in the Farey fan of order 6.
 The first four properties are from [9] and the reader is invited to consult this
 reference for the proofs, that are fairly simple.

Property 4 ([9]). *The abscissas of intersections of a ray $R(x, y)$ of \mathcal{F}_n with
 other rays are **consecutive terms** of a Farey series of order $\max(x, n - x)$.*

In Figure 5(b), the abscissas of the intersections between the ray $R(2, 1)$,
 depicted in red, and the other rays of \mathcal{F}_6 are consecutive terms of the Farey
 series of order $4 = \max(2, 6 - 2)$.

Property 5 ([9]). *Let f_i and f_{i+1} be two consecutive fractions of the Farey
 series of order n . In the interval $f_i < \alpha < f_{i+1}$, there is no intersection of rays.
 Thus, in this interval the Farey fan is a simple **ladder** of rungs.*

In Figure 5(b), two ladders are depicted in blue for $f_i = \frac{1}{3}$ and $f_i = \frac{2}{3}$.

Property 6 ([9]). *Let $v(\frac{p}{q}, \frac{r}{q})$, $0 \leq p \leq q \leq n$, be a point of \mathcal{F}_n . Let $R(x_0, y_0)$
 be the ray of minimum slope passing through v . The other rays passing through
 v have a slope equal to $x_0 + kq$ with $k \in \mathbb{Z}$ and $x_0 + kq \leq n$.*

In Figure 5(b), three rays go through the point $(\frac{1}{2}, \frac{1}{2})$ (in orange). The slopes
 of these rays are equal to $x_0 = 1, 3$ and 5 . From this property, we can derive
 the following corollary.

Corollary 1. *Let $v(\frac{p}{q}, \frac{r}{q})$, $0 \leq p \leq q \leq n$, be a point of \mathcal{F}_n . Let $R(x, y)$ be a
 ray passing through p . R is the ray of smallest slope passing through v if and
 only if $x - q < 0$. It is the ray of greatest slope passing through v if and only if
 $x + q > n$.*

The following property is similar to Corollary 1 in [9], but brings in more
 information.

Property 7. *Let $\frac{p}{q}$ be a fraction of the Farey series of order n . The intersection
 between the line $\alpha = \frac{p}{q}$ and \mathcal{F}_n is exactly the set of points $(\frac{p}{q}, \frac{r}{q})$ where r takes
 all the integer values between 0 and q .*

PROOF. We study the intersection between $R(x, y)$ defined by the equation
 $\beta = -\alpha x + y$ and $\alpha = \frac{p}{q}$. We get $\beta = \frac{-px + qy}{q}$. For $0 \leq y \leq x \leq q \leq n$, the
 quantity $-px + qy$ takes all the integral values in the interval $[0, q]$, which ends
 the proof.

In Figure 5(b), the intersection between $\alpha = \frac{4}{5}$ (depicted in green) and \mathcal{F}_n
 is the set of points $(\frac{4}{5}, \frac{r}{5})$ with $r \in \mathbb{Z}$, $0 \leq r \leq 5$. Using Properties 5 and 7, we
 can prove the following result to compute the ray of smallest slope in a given
 point.

394 **Corollary 2.** Let $v(\frac{p}{q}, \frac{r}{q})$, $0 \leq p \leq q \leq n$, be a point of \mathcal{F}_n . Let $\frac{p'}{q'}$ be the
395 fraction following $\frac{p}{q}$ in the Farey series of order n . The ray of smallest slope
396 passing through v is defined by the point v and the point of coordinates $v'(\frac{p'}{q'}, \frac{r'}{q'})$
397 where r' is such that $\frac{r'}{q'} \leq \frac{r}{q}$ and $\frac{r'+1}{q'} > \frac{r}{q}$.

398 PROOF. From Property 5, \mathcal{F}_n is a ladder in the interval $[\frac{p}{q}, \frac{p'}{q'}]$, which means
399 there is no intersection of rays in this interval. From Property 7, all the rays
400 passing through v cut the line of equation $\alpha = \frac{p'}{q'}$ in a point $v'(\frac{p'}{q'}, \frac{r'}{q'})$, $r' \in \mathbb{Z}$,
401 $0 \leq r' \leq q'$. Among all these rays, the ray of smallest slope is the one that
402 passes through the point $v_{max}(\frac{p'}{q'}, \frac{r_{max}}{q'})$ where r_{max} is the maximal value of r'
403 such that $\frac{r'}{q'} \leq \frac{r}{q}$. It remains to prove that the two points v and v_{max} define a
404 ray of \mathcal{F}_n . Let x and y respectively be the slope and the intercept of the line
405 defined by v and v_{max} . We have to prove that: i) x and y are integers, ii) x is
406 lower than n . For any r' , and from a direct computation, we get $x = \frac{rq' - r'q}{p'q - pq'}$.
407 Since $\frac{p}{q}$ and $\frac{p'}{q'}$ are consecutive fractions of a Farey series, we have $p'q - pq' = 1$
408 and x is an integer. The same relation is used to show that y is an integer, which
409 proves i). Let us prove ii). From the definition of r_{max} we have $\frac{r}{q} - \frac{r_{max}}{q'} < \frac{1}{q'}$,
410 which is equivalent to $rq' - r_{max}q < q$. Since q is lower than or equal to n , this
411 ends the proof.

412 Algorithmically, two solutions are possible to compute the ray of smallest
413 slope through a point using Corollary 2: direct computation or dichotomy. With
414 a direct computation, we get $r_{max} = \lfloor \frac{rq'}{q} \rfloor$: r_{max} is the result of the integer di-
415 vision. A dichotomy costs $\mathcal{O} \log(q')$ and is not interesting since integer numbers
416 only are involved.

417 4.3. Algorithm: a walk in the Farey Fan

418 Following Problem 3, we look for the characteristic point of the facet con-
419 taining a given point $\Lambda(\frac{a}{b}, \frac{r}{b})$. From Proposition 1, Section 4.1 and Property 7
420 we have the following characterization of the characteristic point.

421 **Property 8.** A point $v(\frac{p_v}{q_v}, \frac{r_v}{q_v})$ is the characteristic point of a facet if and only
422 if:

- 423 1. either v is the intersection of the two lower edges:
 - 424 (a) the ray supporting the right lower edge is the one of smallest slope in
425 v ;
 - 426 (b) the ray supporting the left lower edge is the one of greatest slope in v ;
- 427 2. or v is on the unique lower edge and more than one ray passes through the
428 point $(\frac{p_v}{q_v}, \frac{r_v+1}{q_v})$

429 As in [9], the algorithm consists of three steps that are detailed in the fol-
430 lowing sections :

1. Find the ladder to which Λ belongs;
2. Locate the highest ray that lies on or below Λ : this ray supports a lower edge of the facet (Section 4.3.2, Algorithm 5);
3. Walk along the ray(s) to determine the characteristic point (Section 4.3.3, Algorithm 6).

4.3.1. Find the ladder

Given a point $\Lambda(\frac{a}{b}, \frac{\mu}{b})$, finding the ladder to which Λ belongs in \mathcal{F}_n is equivalent to finding the two fractions with a denominator smaller than n closest to $\frac{a}{b}$ (greater and lower). We look for two fractions $f = \frac{p}{q}$ and $g = \frac{p'}{q'}$ such that $q \leq n$, $q' \leq n$, $f \leq \frac{a}{b} \leq g$, and there is no fraction of denominator smaller or equal to n neither between f and $\frac{a}{b}$ nor between $\frac{a}{b}$ and g .

The solution of this problem uses continued fractions representation of the number to be approximated (see [24, 25] for instance for an introduction on continued fractions). The solution of our problem is brought by the following Theorem (stated in [10, 26], with the proof in [27]).

Theorem 1 (as stated in [26]). *Suppose we are required to find the fraction, whose denominator does not exceed n , which most closely approximates, but is no greater than, the quantity $\frac{a}{b}$. If we construct a sequence of fractions containing all the odd principal convergents of $\frac{a}{b}$ with their corresponding intermediate convergents (if such convergents exist), then the fraction we desire is the element of this sequence with the largest denominator no greater than n .*

Thus the fractions we are looking for can be found by searching in the sequence of odd convergents for the greater one, and even convergents for the lower one. A nice geometrical interpretation of the continued fraction of a number was given by Klein in 1895, and can be found in [25] or [28]: if a fraction $\frac{p}{q}$ is represented by the grid point (q, p) , then the odd (resp. even) convergents of a number α are the vertices of the lower (resp. upper) convex hull of the grid points lying above (resp. below) the line $y = \alpha x$. In particular, this leads to a very simple geometrical algorithm to compute the convergents of a rational number. This algorithm called **Geometric-GCD** is presented in [28] and has a complexity $\mathcal{O}(\log(\min(a, b)))$ for a fraction $\frac{a}{b}$.

In order to compute the closest convergent with a bounded denominator, we use **Geometric-GCD** algorithm and simply add an upper bounding constraint of the form $x \leq n$ as in [10] to get the hybrid Algorithm 3. As in [10], the **Intersection**(P, \vec{v}, l) function computes the intersection point between the straight line defined by the point P and the vector \vec{v} , and the straight line l . This point is of the form $P + \alpha \vec{v}$, and the function **Intersection** returns $\lfloor \alpha \rfloor$.

Algorithm 4 implements to computation of the ladder around the fraction $\frac{a}{b}$ in the Farey Fan of order n . If b is greater than n , then it consists in a simple call to the **BoundedGeometricGCD** algorithm. Otherwise, a direct call to the **BoundedGeometricGCD** algorithm would return the fraction $\frac{a}{b}$ for both the lower and the upper fraction, which is not the result sought. However, as shown in Figure 6, since we suppose $b < 2n$, a simple call to **BoundedGeometricGCD** also does the trick.

Algorithm 3: BoundedGeometricGCD(l, n)

l_{right} is the vertical line $x = n$
 $L = (1, 0)$
 $U = (0, 1)$
 $i = 0$ **while** continue **do**
 if i is even **then**
 $\alpha 1 = \text{Intersection}(U, L, l)$, $\alpha 2 = \text{Intersection}(U, L, l_{right})$
 $\alpha = \min(\alpha 1, \alpha 2)$
 $U = U + \alpha L$
 if ($\alpha = \alpha 2$ or U is on l) **then** continue = false
 else
 $\alpha 1 = \text{Intersection}(L, U, l)$, $\alpha 2 = \text{Intersection}(L, U, l_{right})$
 $\alpha = \min(\alpha 1, \alpha 2)$
 $L = L + \alpha U$
 if ($\alpha = \alpha 2$ or L is on l) **then** continue = false
 end
return L as the lower fraction, U as the greater one.

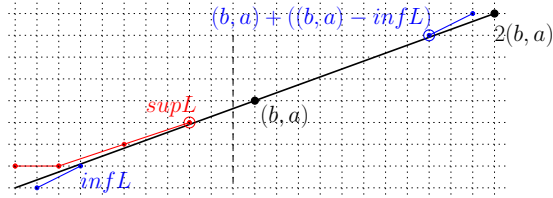


Figure 6: Computation of the ladder when $b \leq n$: the smallest fraction greater than $\frac{a}{b}$ of denominator lower than or equal to n is given by one of the surrounded points.

4.3.2. Locate a lower edge

At this point, we work in a ladder defined by two fractions $f = \frac{p}{q}$ and $g = \frac{p'}{q'}$ of \mathcal{F}_n and $f < g$. This step consists in localising $\mathbf{\Lambda}$ in the ladder by computing the highest ray under $\mathbf{\Lambda}$ in \mathcal{F}_n . In [9], this step is performed as a binary search among the rays of the ladder. However, each stage of the binary search requires to solve a Diophantine equation with the extended Euclidean algorithm, reaching a total complexity of $\mathcal{O}(\log^2 n)$.

Our algorithm, presented in Algorithm 5 and illustrated in Figure 7, also performs a dichotomy (line 3), but only on the rays of smallest slope passing through the points of abscissa $\frac{p}{q}$ (in red in Figure 7). The basic operation used in this part is thus the computation of the position of a point $\mathbf{\Lambda}$ with respect to a given ray: the function `PositionWrtRay(point, ray)` returns `on`, `below` or `above`.

Thanks to Property 7, the set of points of abscissa $\frac{p}{q}$ can be defined as on line 1, and the rays of smallest slope are computed in time $\mathcal{O}(1)$ in the ladder using Corollary 2 (line 2). On line 4, the ray of greatest slope is computed from the ray

Algorithm 4: FindLadder($\frac{a}{b}, n$)

```

     $l : y = \frac{a}{b}x$ 
    1 if  $b > n$  then
         $\frac{p}{q}, \frac{p'}{q'} = \text{BoundedGeometricGCD}(l, n)$ 
    2 else
         $\frac{p}{q} = \frac{a}{b}$ 
         $(\text{inf}L, \text{sup}L) = \text{BoundedGeometricGCD}(l, b - 1)$ 
         $\text{sup}R = (b, a) + ((b, a) - \text{inf}L)$ 
        if ( $\text{sup}L$  closer to  $l$  than  $\text{sup}R$ ) or ( $\text{the abscissa of } \text{sup}R \text{ is greater than } n$ ) then  $\frac{p'}{q'} \leftarrow \text{sup}L$  else  $\frac{p'}{q'} \leftarrow \text{sup}R$ 
    return  $\frac{p}{q}$  and  $\frac{p'}{q'}$ 

```

491 of smallest slope thanks to Property 6: its slope is simply equal to $n - (n - x_0)$
 492 (mod q) where x_0 is the slope of the ray of smallest slope. Property 6 is used in
 493 line 6. Two solutions are possible: (i) either a dichotomy is performed on the
 494 rays passing through v_{j+1} , once again using the function **PositionWrtRay**, (ii)
 495 or a direct computation is done. In the case of (ii), let x be the slope of the
 496 line passing through v_{j+1} and $\mathbf{\Lambda}$. Let $R_{j+1}(x_{j+1}, y_{j+1})$ be the ray of smallest
 497 slope passing through v_{j+1} . Let $[x]$ be the value $x_{j+1} + kq$ nearest to and lower
 498 than x , $k \in \mathbb{Z}$: $[x]$ is equal to $\lfloor x \rfloor + x_{j+1} - (\lfloor x \rfloor \pmod{q})$ if $\lfloor x \rfloor \neq x_{j+1}$, and
 499 equal to x_{j+1} otherwise. If the complexity of solution (i) is straightforwardly
 500 logarithmic in the number of rays, which is smaller than q , the complexity of
 501 solution (ii) is more complicated to evaluate since it depends on the way floor
 502 and modulo functions are implemented. However, we show in Section 4.4 that
 503 dichotomy is of help when floating-point input data is considered.

504 In Figure 7, on the left, the point $\mathbf{\Lambda}$ is located under the ray of greatest slope
 505 passing through v_{j+1} (in green, line 5 in Algorithm 5), R_j is returned. On the
 506 right, the point $\mathbf{\Lambda}$ is in between the rays passing through v_{j+1} .

507 4.3.3. Find the characteristic point

508 Let us denote by M and N the two points defined as the intersection between
 509 the ray $R(x, y)$ returned by Algorithm 5 and the vertical lines defining the
 510 ladder, i.e. $\alpha = \frac{p}{q}$ and $\alpha = \frac{p'}{q'}$ as defined in Section 4.3.1. The segment $[MN]$ is
 511 part of a lower edge of the facet of \mathcal{F}_n containing $\mathbf{\Lambda}$ in \mathcal{F}_n .

512 The first step of the algorithm detailed in Algorithm 6 is to compute the
 513 extremities of the lower edge containing $[MN]$. To do so, the key point is to use
 514 Property 4 to characterize the points of intersection between a ray and other
 515 rays. Given a ray $R(x, y)$ of the Farey Fan \mathcal{F}_n and a point $v(\frac{p}{q}, \frac{r}{q})$ on this ray,
 516 v is the crossing point of several rays if and only if $q \leq \max(x, n - x)$. Thus,
 517 the abscissa of the left (resp. right) extremity of the lower edge is given by the
 518 term of the Farey series of order $\max(x, n - x)$ preceding (resp. following $\frac{p}{q}$
 519 (resp. $\frac{p'}{q'}$) (line 1 of Algorithm 6). This step is simply completed with a call to

Algorithm 5: $\text{localizeLowerEdge}(\frac{p}{q}, \frac{p'}{q'}, \Lambda)$

```

1 Let  $v_i = (\frac{p}{q}, \frac{i}{q})$ ,  $i \in \mathbb{Z}$ ,  $0 \leq i \leq q$ 
2 Let  $R_i(x_i, y_i)$  be the ray of smallest slope passing through  $v_i$ 
3 Perform a dichotomy on the  $R_i$  to compute  $j \in [0, q-1]$  such that  $\Lambda$  is
  above  $R_j$  and below  $R_{j+1}$ 
4 Let  $R'_{j+1}$  be the ray of greatest slope passing through  $v_{j+1}$ 
  if  $\text{PositionWrtRay}(\Lambda, R'_{j+1}) = \text{on}$  then
    return  $R'_{j+1}$ 
  else
5   if  $\text{PositionWrtRay}(\Lambda, R'_{j+1}) = \text{below}$  then
    return  $R_j$ 
  else
6   Among all the rays passing through  $v_{j+1}$ , find the ray  $R$  which is
    right under  $\Lambda$  and return  $R$ 

```

the function **BoundedGeometricGCD** for the line of slope $\frac{p}{q}$ and with an upper bounding constraint set to $\max(x, n-x)$ (Algorithm 6, line 1). We get the two fractions $\frac{p}{q}$ and $\frac{\bar{p}}{q}$ preceding and following $\frac{p}{q}$ in the Farey series of order $\max(x, n-x)$. Note that since $\frac{p}{q}$ and $\frac{p'}{q'}$ are consecutive fractions of the Farey series of order n , the fraction $\frac{\bar{p}}{q}$ is also greater than $\frac{p'}{q'}$. From these two fractions we compute the two points \underline{Q} of R with abscissa equal to $\frac{p}{q}$ and \overline{O} of R with abscissa equal to $\frac{\bar{p}}{q}$ (line 2).

At this point, $[\underline{Q}\overline{O}]$ is a lower edge of the facet containing Λ . Then, the three cases illustrated in Figure 8 can occur: either \underline{Q} or \overline{O} is the characteristic point (case (a) and (b)), or not (case (c)). We use Property 8 to distinguish between these cases:

- if R is the ray of smallest slope in \underline{Q} , then \underline{Q} is the characteristic point: the condition line 3 refers to Corollary 1;
- if R is the ray of greatest slope in \overline{O} , then \overline{O} is the characteristic point: the condition line 4 refers to Corollary 1;
- otherwise, the facet is lower triangular, and the abscissa of the characteristic point is given by the median of the abscissas of the lower edge extremities, *i.e.* \underline{Q} and \overline{O} (direct consequence of Property 4): on line 5, the median is computed, and the point of R with this abscissa is the characteristic point.

4.3.4. General algorithm and Complexity

The general algorithm **FareyFanDSLSubsegment** gathering all the functions presented before is summed up in Algorithm 7. It solves Problem 3, equivalent

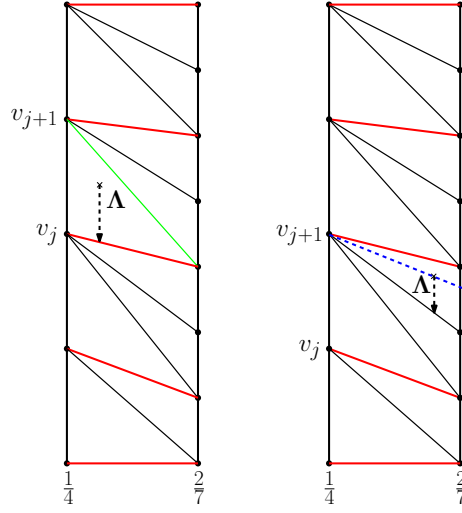


Figure 7: Illustration of Algorithm 5: the dichotomy is performed on the red rays only.

543 to Problem 1 returning the point of the Farey Fan which is the characteristic
 544 point of the DSL subsegment preimage.

545 **Lemma 1.** *The complexity of Algorithm 7 is in $\mathcal{O}(\log(n))$.*

546 **PROOF.** We assume a computing model where standard arithmetic operations
 547 are done in constant time. Finding the ladder is done using Algorithm 4 that
 548 has a complexity of $\mathcal{O}(\log(n))$.

549 The localization of a lower edge is done with Algorithm 5: the computation
 550 of the R_i (line 2) does not have to be done as a precomputation since the
 551 dichotomy (line 3) can be performed on the indices i . Thus they are computed
 552 on the fly and only when necessary, and the complexity of these two lines is in
 553 $\mathcal{O}(\log(q))$ with $q \leq n$. The operations done in lines 4 to 5 are done in constant
 554 time. The complexity of line 6 was discussed in Section 4.3.2 and is $\mathcal{O}(\log(q))$
 555 in the worst case. All in all, the complexity of Algorithm 5 is in $\mathcal{O}(\log(q))$.

556 Algorithm 6 performs the last step of the algorithm. On line 1, computing
 557 the points \underline{Q} and \overline{Q} costs $\mathcal{O}(\log q)$ with $q \leq n$ (see Section 4.3.3). The com-
 558 putation of the mediant fraction on line 5 also has a logarithmic worst time
 559 complexity if the fraction is not irreducible. However, we can show that $\underline{p} + \overline{p}$
 560 and $\underline{q} + \overline{q}$ are relatively prime: since $\frac{\underline{p}}{\underline{q}}$ and $\frac{\overline{p}}{\overline{q}}$ are successive terms of a Farey
 561 series, the denominator of the mediant must be strictly greater than $\max(\underline{q}, \overline{q})$;
 562 if $\frac{\underline{p} + \overline{p}}{\underline{q} + \overline{q}}$ was reducible, there would exist an integer $k \geq 2$ such that $\frac{\underline{p} + \overline{p}}{\underline{q} + \overline{q}} = \frac{k\underline{p}''}{k\underline{q}''}$,
 563 which contradictorily implies $q'' \leq \max(\underline{q}, \overline{q})$. Thus, the mediant computation
 564 is done in constant time.

565 All the other operations of this algorithm take $\mathcal{O}(1)$, which ends the proof.

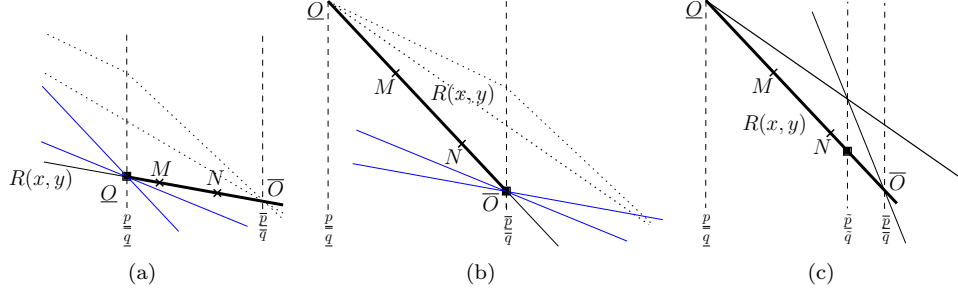


Figure 8: Three cases for the lower edge $[Q\bar{O}]$: (a) all the rays passing through Q (in blue) have a slope greater than x and Q is the characteristic point; (b) all the rays passing through \bar{O} (in blue) have a slope smaller than x and \bar{O} is the characteristic point; (c) neither Q nor \bar{O} is the solution, and the characteristic point is found with a median computation.

Algorithm 6: $\text{findCPoint}(\frac{p}{q}, \frac{p'}{q'}, R)$

Let $R(x, y)$ be the ray output by Algorithm 5

- 1 $(\frac{p}{q}, \frac{\bar{p}}{q}) = \text{BoundedGeometricGCD}(y = \frac{p}{q}x, \max(x, n - x))$
- 2 Let \underline{Q} (resp. \bar{O}) be the intersection point between $\alpha = \frac{p}{q}$ (resp. $\frac{\bar{p}}{q}$) and R
- 3 **if** $x - q < 0$ **then**
 return \underline{Q} **else**
- 4 **if** $x + \bar{q} > n$ **then**
 return \bar{O}
 else
- 5 Let $\frac{\tilde{p}}{\tilde{q}} = \frac{p + \bar{p}}{q + \bar{q}}$
 return the intersection point between $\alpha = \frac{\tilde{p}}{\tilde{q}}$ and R

566 This algorithm solves Problem 3 in $\mathcal{O}(\log(n))$ where n is the order of the
 567 Farey fan. From the equivalence of Problems 1 and 3, this algorithm also solves
 568 Problem 1 in logarithmic time where n is the length of the DSS.

569 4.4. Extensions

570 4.4.1. 4-connected DSL subsegment

571 In the framework presented above, the DSL and DSS considered are 8-
 572 connected sets of pixels. In [8, 1] the authors consider the same problem but
 573 with 4-connected digital straight lines and segments. Their definition is similar
 574 to the 8-connected lines: a 4-connected DSL of integer characteristics (a, b, μ)
 575 is the infinite set of digital points $(x, y) \in \mathbb{Z}^2$ such that $0 \leq ax - by + \mu < a + b$
 576 assuming that, as before, $0 \leq a \leq b$.

577 Adapting the algorithm `FareyFanDSLSubsegment` for the computation of the
 578 minimal characteristics of 4-connected DSL subsegments is actually very easy
 579 using the following property.

Algorithm 7: FareyFanDSLSubsegment($a_{\mathbf{L}}, b_{\mathbf{L}}, \mu_{\mathbf{L}}, P, Q$)

Let $\mathbf{\Lambda} = (\frac{a_{\mathbf{L}}}{b_{\mathbf{L}}}, \frac{\mu_{\mathbf{L}}}{b_{\mathbf{L}}})$
 Let $n = x_Q - x_P$
 $(\frac{p}{q}, \frac{p'}{q'}) = \text{FindLadder}(\frac{a_{\mathbf{L}}}{b_{\mathbf{L}}}, n)$
 $R = \text{localizeLowerEdge}(\frac{p}{q}, \frac{p'}{q'}, \mathbf{\Lambda})$
 $CPoint = \text{findCPoint}(\frac{p}{q}, \frac{p'}{q'}, R)$
return $CPoint$

580 **Property 9.** *The grid point $(x + y, y)$ belongs to the 8-connected DSL $(a, b +$
 581 $a, \mu)$ if and only if the grid point (x, y) belongs to the 4-connected DSL (a, b, μ) .*
 582

583 **PROOF.** $(x + y, y)$ belongs to the 8-connected DSL $D(a, b + a, \mu)$ is equivalent
 584 to $0 \leq a(x + y) - (b + a)y + \mu < b + a$. Rewriting this equation we get $0 \leq$
 585 $ax - by + \mu < b + a$, which is equivalent to say that (x, y) belongs to the
 586 4-connected DSL (a, b, μ) .

587 Thus, a simple shear transform of matrix $M = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ transforms the points
 588 of a 4-connected DSL into a 8-connected DSL as illustrated in Figure 9.

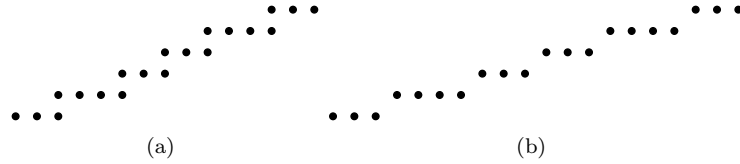


Figure 9: The 4-connected DSL $(3, 7, 0)$ in (a) is transformed into the 8-connected DSL $(3, 10, 0)$

589 Consider now Problem 4 which is the same as Problem 1 for 4-connected
 590 DSL.

591 **Problem 4.** Given a 4-connected DSL L of characteristics $(a_{\mathbf{L}}, b_{\mathbf{L}}, \mu_{\mathbf{L}})$ and two
 592 points $P(x_P, y_P)$ and $Q(x_Q, y_Q)$ of this DSL, compute the minimal character-
 593 istics (a, b, μ) of the DSS $S = \{(x, y) \in L \mid x_P \leq x \leq x_Q\}$.

594 Thanks to Property 9, solving Problem 4 is equivalent to solving Problem
 595 1 for a 8-connected DSL of characteristics $(a_{\mathbf{L}}, b_{\mathbf{L}} + a_{\mathbf{L}}, \mu_{\mathbf{L}})$ and the two points
 596 $P_t(x_P + y_P, y_P)$ and $Q_t(x_Q + y_Q, y_Q)$. If (a, b, μ) is the solution for Problem 1
 597 on this data, then $(a, b - a, \mu)$ is the solution of Problem 4.

4.4.2. When the DSL characteristics are floating-point numbers

599 Let us now consider the case when the characteristics of the DSL given as
 600 input data are not rational numbers anymore, but real numbers. We now have
 601 $\mathbf{\Lambda}(\alpha_{\mathbf{L}}, \beta_{\mathbf{L}})$ with $\alpha_{\mathbf{L}}$ and $\beta_{\mathbf{L}}$ in \mathbb{R} .

602 From a theoretical point of view, the algorithm `FareyFanDSLSubsegment`
603 proposed in Section 4 works the same. However, as often for geometrical al-
604 gorithms, things get more complicated when the implementation is concerned.
605 Working on this issue is a research domain by itself, and a huge literature can
606 be found about robustness in geometrical problems. See for instance [29, 30].

607 Without going to deep in these considerations, we propose a solution to get
608 a robust algorithm for floating-point input data. The robustness is evaluated in
609 Section 5.2.

610 The only functions that may cause some problems are the ones involving
611 directly the point $\Lambda(\alpha_L, \beta_L)$. A careful analysis of the algorithm shows that
612 this concerns only two particular functions: the function `Intersection`(P, \vec{v}, l)
613 called in Algorithm 4.3.1 and the function `PositionWrtRay`(`point`, `ray`) called
614 on lines 4 and 5 of Algorithm 5. In the first case, the computation of the position
615 of an integer point with respect to a line with floating-point characteristics in
616 involved, while in the second case, the computation of the position of a point
617 with floating-point coordinates with respect to a line with integer characteristics
618 is done. Let us also come back to the line 6 of Algorithm 5. We saw in Section
619 4.3.2 that two algorithmic choices are possible: either (i) a dichotomy using the
620 function `PositionWrtRay` or (ii) a direct computation involving the computation
621 of the slope of a line going through Λ . As we see in the few next lines, it is
622 possible to make the function `PositionWrtRay` robust, while the computation
623 of a slope involving floating-point coordinates plus its rounding seems more
624 difficult to control, on an uncertainty point of view. Thus choice (i) seems to
625 be the better one for floating-point input data.

626 The uncertainty over the floating-point data is handled in a very classical
627 way using an ε parameter. The way this parameter is used is illustrated in
628 Figure 10. In (a), a centered band of height ε is defined around the line l of
629 the `Intersection`(P, \vec{v}, l) algorithm. If $P + \alpha\vec{v}$ lies in the gray area, the point
630 is said to be on the line. If it is above the gray area, the point lies above l ,
631 and below otherwise. Similarly, in (b) a vertical interval of height ε is defined
632 around Λ in the function `PositionWrtRay`: if the ray R crosses the interval, the
633 point is said to be on the ray, if R is below the interval, R is below l , and above
634 otherwise. We could equivalently have represented the uncertainty around the
635 ray R in (b) as we do in (a), but since the uncertainty is carried by the point Λ
636 we find this representation more accurate.

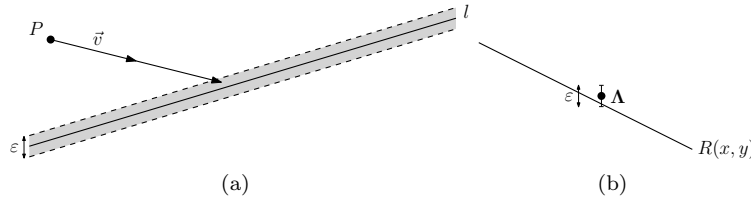


Figure 10: Use of the ε parameter for the `Intersection` (a) and the `PositionWrtRay` (b) functions.

637 The value of the parameter ε depends on the precision of the input data. If
638 the coordinates of $\mathbf{\Lambda}$ are known with a precision of 10^{-r} , then one can ex-
639 pect that the result of the algorithm on the floating-point point $\mathbf{\Lambda}(\alpha_{\mathbf{L}}, \beta_{\mathbf{L}})$
640 is the same as the result obtained for the integer characteristics $(a, b, \mu) =$
641 $(\alpha_{\mathbf{L}}.10^r, 10^r, \beta_{\mathbf{L}}.10^r)$. This suggests that a good value for ε could be 10^{-r} . In-
642 deed, consider a line of equation $l : ax - 10^r y + \mu = 0$, where a and 10^r are
643 relatively prime. Then there is no integer point in the centered band of vertical
644 height $2 * 10^r$. Thus, for instance in the case of the `Intersection` function, any
645 integer point that is in the band of vertical height 10^r is closer from l than from
646 any other line with the same slope and integer characteristics. A similar reason-
647 ing can be done for the function `PositionWrtRay`. An experimental validation
648 is conducted in Section 5.2.

649 Last, we would like to address rapidly the consequences in the case of failures.
650 Concerning the function `PositionWrtRay`, an erroneous answer leads to a bad
651 localization of the lower edge: the result is either the edge below or the edge
652 above the ground truth edge. This means that the final answer will be the
653 characteristic point of the cell just below or just above the ground truth cell.
654 Nevertheless, from the definition of the preimage of a DSS, it is easy to see
655 that the difference between two DSSs the preimage of which share an edge is of
656 exactly one pixel. The abscissa of this pixel is given by the slope of the common
657 edge. This means that an erroneous answer of the function `PositionWrtRay`
658 leads to a difference of at most one pixel between the DSS computed and the
659 ground-truth DSS.

660 Concerning the function `Intersection`, the analysis is more complicated.
661 Indeed, if the two fractions returned by the function are not consecutive fractions
662 in a Farey series, the algorithm fails to find a solution. If the two fractions are
663 not the correct ones, but are consecutive terms of a Farey series the algorithm
664 will output a result, but the error committed is difficult to estimate. A deeper
665 study could be done if need be.

666 5. Experimentation

667 5.1. Implementation and settings

668 The two algorithms presented in this paper are implemented and available
669 in the generic C++ open-source library `DGtal` [31]. The `DGtal` library includes
670 several Digital Straight Segment recognition algorithms. Moreover, the authors
671 of [7, 8, 1] made there algorithms available in this library. Comparing our
672 respective results was then an easy and robust task.

673 To conduct the experiments detailed below, we also reuse the protocol de-
674 scribed in [1] and available as a test file in `DGtal`. The overall protocol is
675 governed by two parameters : N governs the value of b while n is the length of
676 the subsegment. We recall here this protocol that includes a few minor changes.

677 1. Input characteristics (a, b, μ) are randomly chosen as follows:

- 678 • b is randomly chosen in the interval $[N - \frac{N}{2}, N + \frac{N}{2}]$;

- 679 • a is randomly chosen in the interval $[0, b]$ - we also ensure that
680 $\gcd(a, b) = 1$;
- 681 • μ is randomly chosen in the interval $[0, 2N]$;
- 682 2. Points P and Q defining the subsegment are chosen as follows:
- 683 • x_P is randomly chosen in the interval $[0, n]$;
- 684 • x_Q is equal to $x_P + n$ where n is the length of the subsegment;

685 Concerning the number of draws, we randomly draw 4000 couple of values
686 for b and a , for which 5 values of μ are chosen. For each of the 20000 triplets
687 (a, b, μ) , we draw 10 random values for x_P and x_Q .

688 We repeat this process for values of N equal to 10^k with $k \in [1, 9]$. For each
689 N , the values of n are in the interval $[10, 2N]$ with an increment $n \leftarrow \frac{4}{3}n$. Thus,
690 for each couple of values (N, n) we proceed to $4000 * 5 * 10 = 200000$ draws.
691 The mean computation time of these draws for each couple (N, n) is reported.

692 As stated in Section 2.3, easy cases are withdrawn: when n gets bigger and
693 close to N , the number of easy cases increases and the mean time would decrease
694 if they were kept, bringing no useful information on the efficiency of the core of
695 the algorithms.

696 5.2. Experimental correctness

697 The first experiment is to validate the correctness of the implementation
698 of our two algorithms. To do so, many tests have been conducted. First, the
699 results of the algorithms **FareyFanDSLSubsegment** and **localCHDSLSubsegment**
700 have been directly compared to the results given by the linear-in-time algorithm
701 **ArithmeticalDSS**, as implemented in the **DGtal** library.

702 Next, the results of the algorithm **FareyFanDSLSubsegment** in the case of
703 4-connected DSLs (see Section 4.4) have been compared to the results returned
704 by the algorithm **ReversedSmartDSS**.

705 Finally, the correctness of the floating-point implementation of the algorithm
706 **FareyFanDSLSubsegment** was also evaluated. To do so, we reused the protocol
707 defined in the previous section, but the possible values of b were powers of 10.
708 For each $b = 10^k$, random integer values of a and μ were chosen. Then the
709 results of the algorithm on the integer data (a, b, μ) and on the floating-point
710 (decimal) data $(\frac{a}{b}, \frac{\mu}{b})$ were compared. In the experiments, the ε parameter was
711 set to 10^k as explained in Section 4.4, and no errors were reported for the several
712 millions of tests carried out.

713 5.3. General speed contest

714 The three algorithms **ReversedSmartDSS**, **localCHDSLSubsegment** and **FareyFanDSLSubsegment**
715 all have a theoretical logarithmic time complexity, the logarithm being applied
716 to different values (length of the segment, or difference of depth of the input and
717 output continued fractions). Algorithm **SmartDSS** has a time complexity which
718 depends on the sum of the quotients of the continued fraction of the output slope
719 and on the number of pattern repetitions. In [1], the authors showed that the

algorithm **SmartDSS** was always slower than **ReversedSmartDSS**. Consequently,
for the sake of clarity, we compare our algorithms with **ReversedSmartDSS** only
and try to exhibit in which cases one algorithm may be faster than the others.

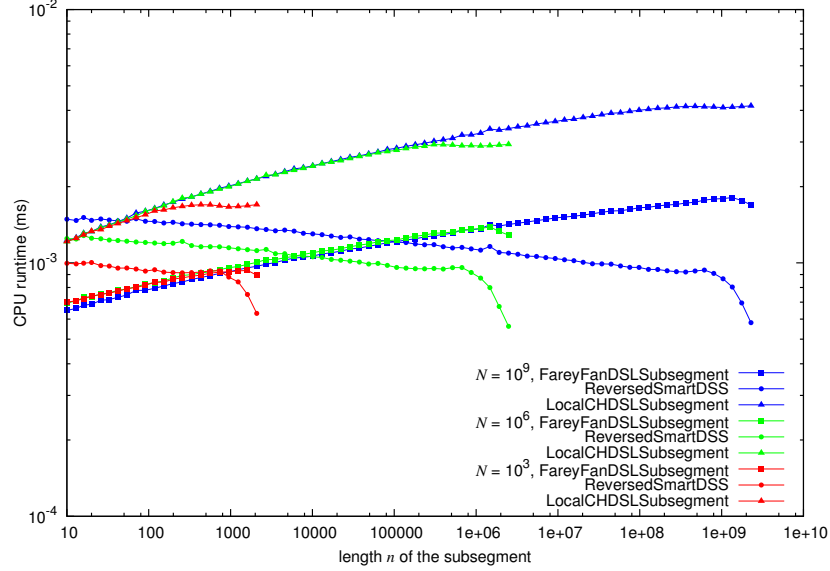


Figure 11: Runtime comparison of our algorithms and one algorithm of [7, 8, 1]

Figure 11 presents the results obtained for three values of N for the three algorithms, for n varying as defined in the previous section. Each value of N is represented by a color (red for 10^3 , green for 10^6 and blue for 10^9) and each algorithm is identified by a point type (square for **FareyFanDSLSubsegment**, disk for **ReversedSmartDSS** and triangle for **localCHDSLSubsegment**).

First if we have a look at a particular value of N (one color), we note that **FareyFanDSLSubsegment** is faster than the other two for small values of n : when n becomes bigger **ReversedSmartDSS** gets better. For very big values of N and very small n , **localCHDSLSubsegment** is faster than **ReversedSmartDSS** but still slower than **FareyFanDSLSubsegment**.

Next, this graph also brings information about the behaviour of each algorithm for increasing values of N . Algorithms **FareyFanDSLSubsegment** and **localCHDSLSubsegment** seem to be insensitive to the value of N for small values of n : for a given n , the computation time is similar for all N . However, in both cases, a slight decrease of the mean computation time occurs when n gets bigger than N .

Concerning the algorithm **ReversedSmartDSS**, the graph reflects the fact that the complexity depends on both the value of N and the value of n : for a given n , the lower the N , the faster the computation.

To conclude on this experimental study, let us replace this work in the context of image analysis, where the DSS length is bounded by the image size.

744 Considering that best compact consumer cameras provide 10 megapixels im-
 745 ages, the maximal length of DSSs in such images is bounded by a few thousands
 746 of pixels. If gigapixel images are considered, the length of the DSSs can reach
 747 values of several tens of thousands of pixels, not more for now. In both cases,
 748 our algorithm **FareyFanDSLSubsegment** is very competitive for any value of N .

749 6. Conclusion

750 We have proposed two algorithms to compute the characteristics of a DSS
 751 which is a subsegment of a DSL of known characteristics. These algorithms
 752 use two dual representations of the set of separating lines for a given set of
 753 grid points. **localCHDSLSubsegment** uses local computation of upper and lower
 754 convex hull to find the separating line of minimal characteristics. We provide
 755 the theoretical proof that only a few edges of the hulls are necessary to find
 756 the result. **FareyFanDSLSubsegment** uses the Farey fan and its numerous arith-
 757 metical properties. With this algorithm, the structure of the set of separating
 758 lines does not need to be computed since it is known through the Farey Fan,
 759 and the problem comes down to a point localization in an arrangement. We
 760 also showed that it could be extended to floating-point input data.

761 Both algorithms have a logarithmic time complexity. Moreover, they are
 762 efficient in practice, and easy to implement. The results have been thoroughly
 763 compared to existing algorithms, both in terms of correctness of the result (to
 764 validate the implementation) and in terms of computation time.

765 There now exists four algorithms of logarithmic time complexity to solve
 766 the DSL subsegment problem. However, no lower bound on the complexity has
 767 been proven so far. Is it possible to compute the DSL subsegment minimal
 768 characteristics in sub-logarithmic time ? Is it possible in constant time ? This
 769 is still an open question.

770 Another perspective is to use this algorithm in fast digitization algorithms.
 771 Suppose we want to digitize a straight segment given by its two floating-point
 772 endpoints on a grid of size n . A fast solution could be to compute the min-
 773 imal characteristics of the DSS before drawing it using the arithmetical DSS
 774 definition.

775 References

- 776 [1] J.-O. Lachaud, M. Said, Two efficient algorithms for computing the charac-
 777 teristics of a subsegment of a digital straight line, *Discrete Applied Math-*
 778 *ematics* 161 (15) (2013) 2293–2315.
- 779 [2] B. Kerautret, J.-O. Lachaud, Meaningful scales detection along digital con-
 780 tours for unsupervised local noise estimation, *Pattern Analysis and Ma-*
 781 *chine Intelligence, IEEE Transactions on* 34 (12) (2012) 2379–2392.
- 782 [3] I. Debled-Rennesson, J.-P. Reveillès, A linear algorithm for segmentation
 783 of digital curves, *Inter. Jour. of Pattern Recog. and Art. Intell.* 9 (6) (1995)
 784 635–662.

- 785 [4] A. Troesch, Interprétation géométrique de l'algorithme d'euclide et recon-
786 naissance de segments, *Theor. Comput. Sci.* 115 (2) (1993) 291–319.
- 787 [5] L. Dorst, A. W. Smeulders, Discrete straight line segments: Parameters,
788 primitives and properties, in: *Vision Geometry*, series Contemporary Math-
789 ematics, Am. Math. Soc., 1991, pp. 45–62.
- 790 [6] A. Rosenfeld, R. Klette, Digital straightness: a review, *Discrete Applied*
791 *Mathematics* 139 (1-3) (2004) 197–230.
- 792 [7] M. Said, J.-O. Lachaud, F. Feschet, Multiscale discrete geometry, in: *Dis-*
793 *crete Geometry for Computer Imagery*, Vol. 5810 of *Lect. Notes in Comp.*
794 *Sci.*, Springer, 2009, pp. 118–131.
- 795 [8] M. Said, J.-O. Lachaud, Computing the characteristics of a subsegment of
796 a digital straight line in logarithmic time, in: *Discrete Geometry for Com-*
797 *puter Imagery (DGCI)*, Vol. 6607 of *Lecture Notes in Computer Science*,
798 Springer, 2011, pp. 320–332.
- 799 [9] M. D. McIlroy, A Note on Discrete Representation of Lines, *AT&T Tech-*
800 *anical Journal* 64 (2) (1985) 481–490.
- 801 [10] E. Charrier, L. Buzer, Approximating a real number by a rational num-
802 ber with a limited denominator: A geometric approach, *Discrete Applied*
803 *Mathematics* 157 (16) (2009) 3473 – 3484.
- 804 [11] T. Roussillon, Algorithmes d'extraction de modèles géométriques discrets
805 pour la représentation robuste des formes, Ph.D. thesis, Universit Lumire
806 Lyon 2 (2009).
- 807 [12] I. Sivignon, Walking in the farey fan to compute the characteristics of a
808 discrete straight line subsegment, in: *International Conference on Discrete*
809 *Geometry for Computer Imagery*, Vol. 7749 of *Lecture Notes in Computer*
810 *Science*, Springer Berlin Heidelberg, 2013, pp. 23–34.
- 811 [13] I. Sivignon, F. Dupont, J.-M. Chassery, Digital intersections : minimal
812 carrier, connectivity and periodicity properties., *Graphical Models* 66 (4)
813 (2004) 226–244.
- 814 [14] T. Roussillon, L. Tougne, I. Sivignon, Robust decomposition of a digital
815 curve into convex and concave parts, in: *19th International Conference on*
816 *Pattern Recognition*, 2008, pp. 1–4.
- 817 [15] L. Dorst, A. N. M. Smeulders, Discrete representation of straight lines,
818 *IEEE Transactions of Pattern Analysis and Machine Intelligence* 6 (4)
819 (1984) 450–463.
- 820 [16] D. Coeurjolly, I. Sivignon, F. Dupont, F. Feschet, J.-M. Chassery, On
821 digital plane preimage structure, *Discrete Applied Mathematics* 151 (1–
822 3) (2005) 78–92.

- [17] J. O'Rourke, An on-line algorithm for fitting straight lines between data ranges, *Commun. ACM* 24 (9) (1981) 574–578.
- [18] B. Hayes, On the Teeth of Wheels, *American Scientist* 88 (4) (2000) 296–300.
- [19] R. Klette, A. Rosenfeld, *Digital geometry geometric methods for digital picture analysis*, Morgan Kaufmann, 2004.
- [20] J. Koplowitz, M. Lindenbaum, A. Bruckstein, The number of digital straight lines on an $n \times n$ grid, *IEEE Trans. on Info. Theory* 36 (1) (1990) 192–197.
- [21] C. Berenstein, D. Lavine, On the number of digital straight line segments, *IEEE Trans. on Pattern Anal. and Mach. Intell.* 10 (6) (1988) 880–887.
- [22] R. Wein, E. Fogel, B. Zukerman, D. Halperin, CGAL - 2D Arrangements. URL http://www.cgal.org/Manual/3.3/doc_html/cgal_manual/Arrangement_2/Chapter_main.html
- [23] V. Kovalevsky, New definition and fast recognition of digital straight segments and arcs, in: *Inter. Conf. on Patt. Anal. and Mach. Intell.*, 1990, pp. 31–34 vol.2.
- [24] G. H. Hardy, E. M. Wright, *An Introduction to the Theory of Numbers*, Oxford Society, 1989.
- [25] H. Davenport, *The Higher Arithmetic: An Introduction to the Theory of Numbers*, Cambridge University Press, 1999.
- [26] W. Harvey, S. J. C. C, Computing two-dimensional integer hulls, *SIAM Journal on Computing* 28 (1999) 28–6.
- [27] G. Chrystal, *Algebra: An Elementary Text-book for the Higher Classes of Secondary Schools and for Colleges*, no. vol. 2, Chelsea Publishing Company, 1964.
- [28] S. Har-peled, An output sensitive algorithm for discrete convex hulls, *Comput. Geom. Theory Appl* 10 (1997) 131–136.
- [29] L. Kettner, K. Mehlhorn, S. Pion, S. Schirra, C. Yap, Classroom examples of robustness problems in geometric computations, *Comput. Geom. Theory Appl.* 40 (1) (2008) 61–78.
- [30] K. Mehlhorn, C. Yap, *Robust Geometric Computation*, to appear.
- [31] DGtal: Digital Geometry Tools and Algorithms Library. URL <http://libdgtal.org>